CLAIMS

What is claimed is:

1.   A method for maintaining data integrity, comprising the computer-implemented
steps of:
     generating checksum data by performing a physical checksum calculation on a
         block of data in volatile memory;
     after generating said checksum data,
         performing a logical check on data contained within the block of data; and
           if the block of data passes said logical check, then causing the block of
             data to be written to nonvolatile memory.

2.   The method of Claim 1 wherein the steps of generating checksum data and
performing a logical check are performed in response to a request to write said
block of data to nonvolatile memory.

3.   The method of Claim 1 further comprising the step of writing the checksum data
to nonvolatile memory in association with writing said block of data to
nonvolatile memory.

4.   The method as recited in Claim 3, further comprising the steps of:
     after writing the block of data to nonvolatile memory,
         causing the block of data and said checksum data to be read from
           nonvolatile memory; and
         performing a physical checksum verification procedure on said block of
           data based on said checksum data, wherein the physical checksum
           verification procedure indicates whether the block of data was
           corrupted subsequent to performing the logical check on the data
           contained with the block of data.

50277-1633 (OID# 2000-135-01)

1   5. The method as recited in Claim 1, further comprising the step of performing one
2   or more physical checksum verification procedures prior to writing the block of
3   data to nonvolatile memory, wherein the one or more physical checksum
4   verification procedures indicate whether the block of data was corrupted
5   subsequent to generating said checksum data.

1   6. The method as recited in Claim 1, wherein:
2   the step of performing a physical checksum calculation comprises the step of a
3   software application performing the physical checksum calculation on
4   said block of data; and
5   the step of performing a logical check on data contained with the block of data
6   comprises the step of said software application performing the logical
7   check on data contained with the block of data.

1   7. The method as recited in Claim 4, wherein:
2   the step of performing a physical checksum calculation comprises the step of an
3   software application performing the physical checksum calculation on
4   said block of data;
5   the step of performing a logical check on data contained with the block of data
6   comprises the step of said software application performing the logical
7   check on data contained with the block of data; and
8   the step of performing a physical checksum verification procedure on said block
9   of data comprises the step of said software application performing the
10   physical checksum verification procedure on said block of data.

1   8. The method as recited in Claim 5, wherein:
2   the step of performing a physical checksum calculation comprises the step of an
3   software application performing the physical checksum calculation on
4   said block of data;

5   the step of performing a logical check on data contained with the block of data

6   comprises the step of said software application performing the logical

7   check on data contained with the block of data; and

8   the step of performing one or more physical checksum verification procedures

9   prior to writing the block of data to nonvolatile memory comprises the

10   step of one or more components other than said software application

11   performing the one or more physical checksum verification procedures

12   prior to writing the block of data to nonvolatile memory.

1   9.   The method as recited in Claim 4, further comprising the step of:

2   after performing the physical checksum verification procedure on said block of

3   data, storing the block of data as a backup version of the block of data,

4   wherein the backup version of the block of data is maintained separate

5   from said block of data in said nonvolatile memory.

1   10.   A method for storing data in a nonvolatile memory, comprising the computer-

2   implemented steps of:

3   determining a desired location in said nonvolatile memory for storing a data

4   block;

5   inserting an address value in said data block, wherein the address value identifies

6   the desired location;

7   prior to performing an operation that stores the data block to nonvolatile memory,

8   verifying that the address value contained within the data block correctly

9   identifies the location in nonvolatile memory into which the operation is

10   going to store the data block; and

11   performing the operation to store the data block to nonvolatile memory only if the

12   address value contained within the data block correctly identifies the

13   location in nonvolatile memory into which the operation is going to store

14   the data block.

1   11.   The method as recited in Claim 10, further comprising the steps of:

2   after storing the block of data to nonvolatile memory,

3   reading the block of data from a location in nonvolatile memory; and

4   comparing the address value contained within the data block with the location in

5   nonvolatile memory from which the data block was read.

1   12.   The method as recited in Claim 10, further comprising the step of maintaining a

2   mapping that identifies a specific location in said nonvolatile memory into which

3   said data block is to be stored.

1   13.   The method as recited in Claim 10, wherein:

2   the step of determining a location in said nonvolatile memory comprises the step

3   of determining a plurality of locations in said nonvolatile memory for

4   storing said data block;

5   the step of inserting an address value in said data block, comprises the step of

6   inserting a plurality of address values in said data block, wherein the

7   plurality of address values identify multiple locations in said nonvolatile

8   memory for which the data block is to be stored; and

9   the step of storing the data block to nonvolatile memory comprises the step of

10   storing the data block in each of the multiple locations in nonvolatile

11   memory only after verifying the plurality of address values includes an

12   address value that correctly identifies the location in nonvolatile memory

13   into which the data block is to be stored.

1   14.   A method for maintaining data integrity, comprising the computer-implemented

2   steps of:

3   performing a physical checksum calculation on a block of data;

4   after performing the physical checksum calculation,

5   performing a first physical checksum verification procedure on said block

6   of data prior to writing the block of data to nonvolatile memory,

50277-1633 (OID# 2000-135-01)

7	wherein the first physical checksum verification procedure
8	indicates whether the block of data was corrupted subsequent to
9	performing the physical checksum calculation on the data
10	contained with the block of data; and
11	if the block of data passes said first physical checksum verification
12	procedure, then causing the block of data to be written to
13	nonvolatile memory.

1 15. The method as recited in Claim 14, further comprising the steps of:

2 after writing the block of data to nonvolatile memory,

3 causing the block of data to be read from nonvolatile memory; and

4 performing a second physical checksum verification procedure on said

5 block of data, wherein the second physical checksum verification

6 procedure indicates whether the block of data was corrupted

7 subsequent to performing the first physical checksum verification

8 procedure on the data contained with the block of data.

1 16. The method as recited in Claim 14, wherein the step of performing a first physical

2 checksum verification procedure includes the steps of performing a plurality of

3 physical checksum verification procedures on said block of data prior to writing

4 the block of data to nonvolatile memory, wherein the plurality of physical

5 checksum verification procedures indicate whether the block of data was

6 corrupted subsequent to performing the physical checksum calculation on the

7 data contained with the block of data.

1 17. The method as recited in Claim 14, wherein:

2 the step of performing a physical checksum calculation comprises the step of a

3 software application performing the physical checksum calculation on

4 said block of data; and

5 the step of performing a first physical checksum verification procedure on said

6 block of data comprises the step of a component other than said software

-31-

7   application performing said first physical checksum verification procedure

8   on said block of data prior to writing the block of data to nonvolatile

9   memory.

1 18.   The method as recited in Claim 17, wherein the step of performing a physical

2   checksum verification procedure on said block of data comprises the step of a

3   disk array component performing the physical checksum verification procedure

4   on said block of data, wherein the disk array component is configured to write the

5   block of data to disk only after verifying the integrity of the data block.

1 19.   The method as recited in Claim 14, further comprising the step of:

2   after performing the physical checksum calculation,

3   performing a logical check on data contained with the block of data; and

4   if the block of data does not pass said logical check, then not writing the

5   block of data to nonvolatile memory.

1 20.   The method as recited in Claim 19, further comprising the step:

2   the step of performing a physical checksum calculation comprises the step of an

3   software application performing the physical checksum calculation on

4   said block of data;

5   the step of performing a logical check on data contained with the block of data

6   comprises the step of said software application performing the logical

7   check on data contained with the block of data; and

8   the step of performing a first physical checksum verification procedure prior to

9   writing the block of data to nonvolatile memory comprises the step of one

10   or more components other than said software application performing one

11   or more physical checksum verification procedures prior to writing the

12   block of data to nonvolatile memory.